# 基于 FPGA 的万兆以太网 UDP_IP 硬件协议栈设计与实现 *

董永吉，王　钰，袁　征
(解放军战略支援部队信息工程大学，郑州 450002)

**摘　要：**针对传统的基于软件协议栈无法满足高速数据传输处理的需求，提出了一种基于硬件加速的 UDP 协议栈设计方案，该方案基于硬件高效并行的特点，实现了 UDP/IP 协议栈，满足了万兆以太网数据传输高带宽需求的问题，通过实际测试表明，该设计最高可以达到 9.32Gbps 传输速率，满足 10Gbps 带宽下线速处理的需求，与传统软件实现相比，处理能力更接近理论极限。

**关键词：**FPGA；万兆以太网；硬件协议栈；UDP 协议
**中图分类号：**TP393　　doi: 10.19734/j.issn.1001-3695.2021.12.0689

## Design and implementation of 10G ethernet UDP/IP hardware protocol stack based on FPGA

Dong Yongji, Wang Yu, Yuan Zheng
(*PLA Strategic support force information engineering university*, *Zhengzhou* 450002, *China*)

**Abstract:** Aiming at the problem that the traditional software protocol stack can not meet the performance requirement of high-speed data transmission, this paper presents a design scheme of UDP protocol stack based on hardware acceleration. The design is based on the characteristics of efficient and parallel hardware to implements the UDP/IP protocol stack. and solves the problem of poor performance of high-speed data transmission. The actual test shows that the design can reach the highest level. To 9.32 Gbps transmission rate, it meets the need of 10 Gbps bandwidth downlink speed processing. Compared with traditional software implementation, it shows that the processing capacity of the proposed method is closer to the theoretical limit.

**Key words:** FPGA; 10G ethernet; hardware protocol stack; UDP protocol



## 0　引言

随着网络技术的飞速发展、网络带宽的迅速提升，数以万计的视频、图像等业务数据逐渐成为高速链路传输的主要组成，面对海量数据的传输压力，传统操作系统内置协议栈或以软件为核心的协议栈加速技术已经不能满足高速、低延迟传输的需求。

根据网络处理经验，1Gbps 的以太网传输需要 1GHz 的处理器频率，面对万兆高速网络的传输需求，大量 CPU 计算资源都被将消耗网络协议的处理上。基于硬件实现的 UDP/IP 协议栈，因通过硬件固化协议栈的处理逻辑，进而能提供高吞吐率、低延迟、高带宽的传输性能[1]。

鉴于 FPGA 兼备灵活可配置和高速并行处理的特性，尤其适合在高速传输领域固化协议栈来提升系统的处理能力。熊光阳[2]等人提出了一种基于 FPGA 实现 MAC 及 UDP/IP 协议栈的方法，旨在解决千兆以太网高速传输的问题；崔鹤[3]等利用硬件实现了 UDP/IP 协议数据的封装和拆封；Guixé Orriols G[4]面向 DAQ 应用，基于 INTEL FPGA 实现了一个通用的千兆 UDP/IP 协议核；冯琛[5]面向高性能协议协议处理需求，设计了一个高效的千兆协议栈；Nianyun Liu[6]等人提出了一个适用于大型传感器网络的 UDP/IP 协议栈，比软件处理提升了 8 倍的性能。面对万兆应用场景，王禹衡[7]设计了一个 10G 以太网 UDP/IP 处理器视频传输接口；夏杨[8]设计了一种万兆以太网分发平台，满足万兆以太网 UDP 传输的需求；彭鹏[9]面向核物理低时延的处理需求，实现了一个高速协议处理模块，而国外厂商 PLDA[10]、Fraunhofer HHI[11]、Intilop[12]以及 Dini Group[13]等公司也分别提出价格不菲的商用 IP 核。

综上可知，当前硬件协议栈研究主要集中在千兆以太网环境，而万兆以太网的硬件协议栈是当前的应用热点。本文基于 FPGA 实现的 UDP/IP 协议栈为面向 10G 网络环境设计，具有以下优势：1.集协议帧的解析、处理、发送和接收于一体，大幅提升系统 UDP 应用的传输效能。2.基于 FPGA 可重构可配置的特点，可以部署于不同场景的网络环境中；3.面向服务器应用设计，可支持多端口多点业务的并发连接。

## 1　硬件结构设计思路

为了提高 UDP 协议的传输效能，本文提出一种基于 FPGA 的 UDP 协议栈设计方案，该设计基于斯坦福大学的 NetFPGA-10G[14]板卡实现，能够尽可能多地支持各种应用。本方案利用该板卡 PCI Express 接口的便携扩展能力强的特点，方便在服务器应用场景中通过扩展物理接口的方式，部署该硬件协议栈到服务器系统中，提升服务器中各种应用 UDP 协议的加速传输处理。

FPGA 作为硬件子系统设计中的重点和难点，根据 FPGA 硬件结构特点实现了协议处理功能，并将硬件部分划分为 10GMAC 接口模块、协议解析模块、ARP 响应模块、ICMP 响应模块、协议封装模块、UDP 协议处理模块、PCIE-DMA 模块，模块划分如图 1 所示。

1)协议解析模块

该模块主要实现数据报文的协议解析处理，并为后续响应报文的生成和封装提取用户的相关信息。首先该模块对到达系统的数据报文协议逐层进行解析，最终实现对本地支持的 UDP 协议请求协议报文的解析、本地协议报文的提取以

**收稿日期：**2021-12-31；**修回日期：**2022-02-28　**基金项目：**国家重点研发计划项目(2019YFB1802502，2019YFB1802500)

**作者简介：**董永吉(1983-)，男，吉林汪清人，副研究员，博士，主要研究方向为高速网络技术（ug_dong@qq.com）；王钰(1997-)，男，山西介休人，助理工程师，学士，主要研究方向为新型网络；袁征(1982-)，女，河南驻马店人，助理研究员，学士，主要研究方向为新型网络.

及无关报文的过滤三部分功能，鉴于 FPGA 硬件结构的特殊性，无法完整实现所有协议的解析和处理，针对于设计的应用场景，实现的具体处理算法流程如下图 2 所示。

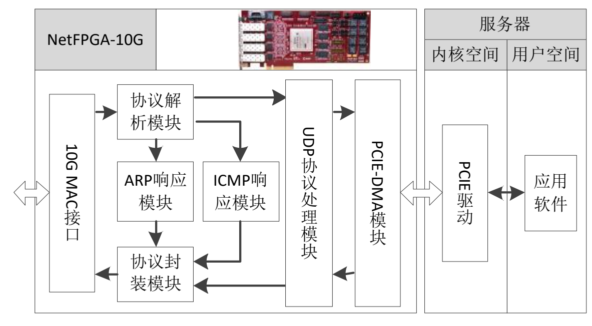



图 1 系统结构组成框图

Fig. 1 System structure composition block diagram

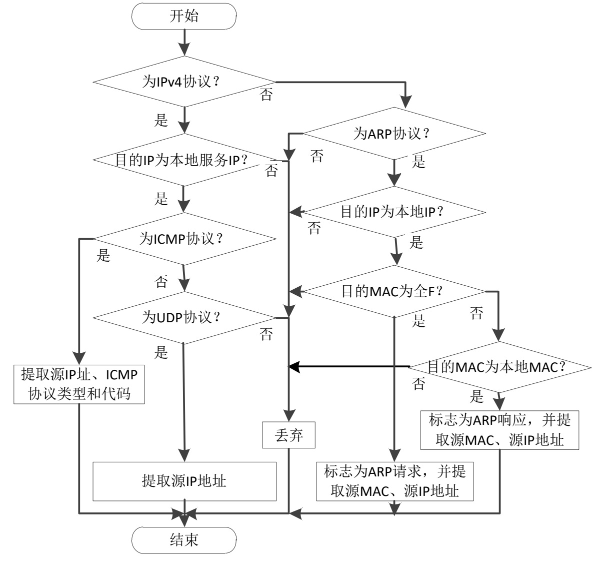



图 2 协议处理流程图

Fig. 2 Protocol processing flowchart

具体处理算法步骤如下所示。

step 1)：判断到达以太网帧的以太类型是否为 0x0800，如果为 0x0800(IPv4 协议)，则执行 step 2，否则执行 step7；

step 2)：判断到达的 IPv4 协议报文中目的 IP 字段是否匹配本地配置的多个服务 IP，如果命中本地多个服务 IP 中的一个，则执行 step 3，否则丢弃该报文；

step 3)：判断到达的 IPv4 协议报文头中协议字段是否为 ICMP，如果为 ICMP 协议，则执行 step 4，否则执行 step 6；

step 4)：提取到达的 ICMP 协议报文中源 IP 地址和 ICMP 协议号，用于后续 ICMP 响应报文的生成，并跳转结束协议解析处理；

step 5)：判断到达的 IPv4 协议报文头中协议字段是否为 UDP，如果为 UDP 协议，则执行 step 6，否则丢弃该报文；

step 6)：提取到达的 UDP 协议报文中源 IP 地址，用于后续 UDP 响应报文的封装，并跳转结束协议解析处理；

step 7)：判断到达以太网帧的以太类型是否为 0x806，如果为 0x0806(ARP 协议)，则执行 step 8，否则丢弃该报文；

step 8)：判断到达的 ARP 协议报文中目的 IP 字段是否匹配本地配置的多个服务 IP，如果命中本地多个服务 IP 中的一个，则执行 step 9，否则丢弃该报文；

step 9)：判断到达数据帧的目的 MAC 地址是否为全 F(广播地址)，如果为全 F，则执行 step 10，否则执行 step 12；

step 10)：判断到达数据帧的目的 MAC 地址是否为本地 MAC，如果是本地 MAC 地址，则执行 step 11，否则丢弃该报文；

step 11)：提取到达的 ARP 协议报文中源 MAC 地址和源 IP 地址，用于构建 ARP 缓存表以及后续 APR 响应报文的重组，并跳转结束协议解析处理；

step 12)：提取到达的 ARP 协议报文中源 MAC 地址和源 IP 地址，用于构建 ARP 缓存表以及后续 ARP 响应报文的生成，并跳转结束协议解析处理。

2)ARP 响应模块

该模块实现 ARP 请求、响应报文的构建以及 ARP 缓存表的维护功能。基于 FPGA 构建该模块时，采用 FPGA 片内 BLOCK RAM 存储 ARP 表项，鉴于硬件资源的有限性，故在标准的 ARP 协议处理的基础上，该模块增加了硬件资源这一限制条件，即当 BLOCK RAM 中存储的 ARP 表即将溢出时，会选择性删除老化时间最长的表项，进而维持整个 ARP 表的正常操作。

首先根据 ARP 请求报文信息，构建 ARP 响应报文；其次响应协议封装模块请求，提供 IP-MAC 映射服务，并定时刷新 ARP 缓存表，根据老化时间删除过期表项；再次根据需求，对于缓存表中未存在的 IP-MAC 映射关系，主动组织 ARP 请求报文内容。ARP 缓存表维护的算法流程如下图 3 所示。

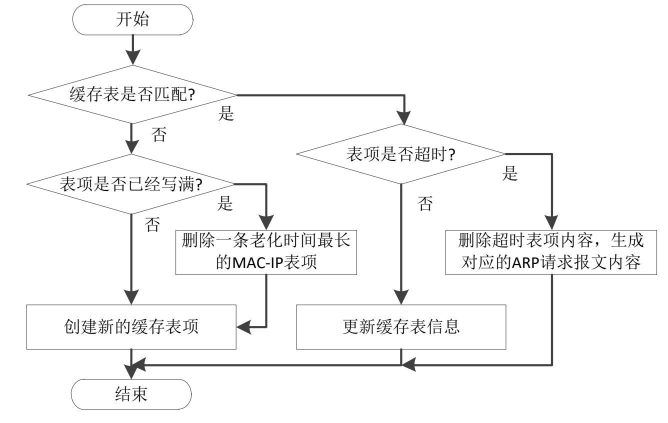



图 3 ARP 缓存表维护算法流程图

Fig. 3 ARP cache table maintenance algorithm flowchart

ARP 缓存表维护具体步骤如下所示。

step 1)：判断使用待发送数据包的目的 IP 地址与缓存表空间中的表项进行匹配，如果匹配，执行 step 2，否则执行 step 3；

step 2)：检查表项空间中的表项地址上是否已经全部建立了缓存表项(表项空间是否已经被使用完毕)，如果是，执行 step 6，否则执行 step 7；

step 3)：依据缓存表中存储的时间戳与当前时刻进行对比，判断待匹配的缓存表项是否已经超时，如果是，执行 step 4，否则执行 step 5；

step 4)：删除超时的缓存表项信息，并生成对应的 ARP 请求报文内容；

step 5)：更新缓存表项的时间戳信息；

step 6)：删除缓存表空间中的一条老化时间最长的表项；

step 7)：在表项空间中未被占用的地址上创建新的缓存表项。

3) ICMP 协议响应模块

该模块实现了 ICMP 协议的一个子集。每当收到 ICMP 请求报文之后，根据当前系统的状态，按照 ICMP 协议规定，对应生成相应的响应报文。本文设计的 UDP 协议栈目标部署在服务器端，故根据服务终端的协议特性，实现了包括“回显应答”、“超时”和“目标不可达”三类响应功能。其中“回显应答”用于支持客户端的 PING 信息，“超时”用于支持数据段 TTL 过期，“目标不可达”用于指示服务器端未开放的端口。

4) 协议封装模块

该模块主要实现两个功能，其一是对 UDP 数据报进行协议封装；其二是对封装后的 UDP 协议数据帧、ICMP 协议数据帧和 ARP 协议数据帧进行数据合流，统一发送到 10G MAC 接口。其中针对 UDP 报进行 IP 协议的封装时，需要根据用户的 IP 地址(目的 IP 地址)信息在 ARP 响应模块的 ARP 缓存表中匹配查找表项，并获取该 IP 地址对应 MAC 地址信息，进而将该 IP 包封装成数据帧，协议封装模块的结构如下图 4 所示。

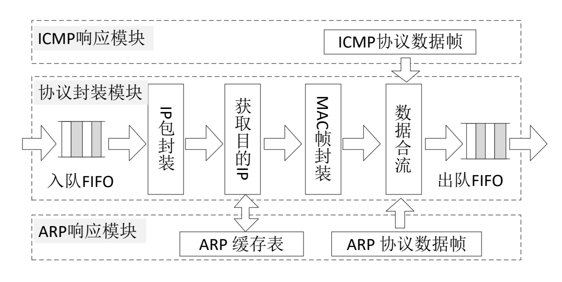



图 4 协议封装模块组成图

Fig. 4　Protocol encapsulation module composition diagram

5)UDP 协议处理模块

该模块主要实现 UDP 协议的封装和拆装，以及 UDP 校验和的计算，以及与应用软件的套接字接口和数据通道。UDP 协议处理模块与应用软件接口如下图 5 所示。

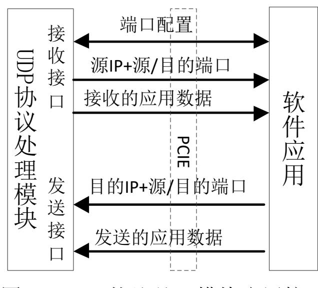



图 5　UDP 协议处理模块应用接口

Fig. 5　UDP protocol processing module application interface

软件应用运行在服务端，软件功能在提供服务之前，需要通过配置端口向硬件协议栈申请/释放服务端口，硬件模块根据服务端口的状态，决定提供或是拒绝服务。

当该模块接收到来自线路的数据时，若通信的目的端口为开放状态，硬件则进行 UDP 协议的拆装，提取通信套接字和接收数据；若通信的目的端口为关闭状态，硬件则丢弃该 UDP 数据段。

当该模块接收到来自软件的数据时，若通信的源端口为开放状态，硬件则根据软件提供的套接字和应用数据，封装成 UDP 数据段；若通信的源端口为关闭状态，硬件则丢弃该应用数据。

为了实现服务器场景下对众多 UDP 端口状态的管理，支持多端口多点业务的并发连接，该模块内部采用双口 BRAM 建立了一个地址为 16bit 位宽的端口状态表，软件系统通过申请/释放操作将该查找表中对应端口号设置为开放/关闭，状态表中共有 $2^{16}$ 个地址，可以将 UDP 所有的端口号一一映射到该表中，且每个表项内有 1 bit 指示位用于标志该端口号是否开放，如下图 6 所示，53 端口(DNS 协议)状态开放。并结合时间和事件触发机制轮询整个状态表，实现对整个 UDP 端口的生存周期管理。通过在 FPGA 片内资源搭建这样一个端口状态查找表，支持多端口状态的并行查找，实现多端口多业务的共存，利用双端口 RAM 的操作特性，也可以有效提升多端口多业务时状态的查找速率。

6)PCIe-DMA 模块

PCIe-DMA 模块主要包括 Xilinx PCIe IP 核和 DMA 模块。PCIe IP 核直接在 Vivado 中调用，用来实现 PCIe 协议的物理层和数据链路层。如图 5 所示，PCIe-DMA 模块主要实现了 UDP 协议处理模块与上位机软件的透明通信。

2^16

| 地址位 | 数据位 |
| --- | --- |
| 0 | 0 |
| 1 | 0 |
| 2 | 0 |
| …… | 0 |
| 53 | 1 |
| …… | 0 |
| 2^16 - 1 | 0 |

图 6　服务端口开放标志

Fig. 6　Schematic diagram of service port open sign

## 2　设计验证

本设计的验证分为功能仿真验证和物理性能验证。验证的硬件平台依托于斯坦福大学的 NetFPGA-10G 板卡，具有 4 个 10Gb/s 的 PCI Express 适配卡，采用 Virtex-5XC5VTX240T-2 FPGA 作为主处理芯片，该 FPGA 可用的逻辑资源为 18,720 可编程逻辑单元(configurable logic blocks)、2,400Kbit 分布式存储单元(Distributed RAM)和 11,664K(324x36K)bit 块存储单元(Block RAMs)，整个系统采用 verilog 语言开发，基于 Xilinx 公司的 Vivado 2016.4 工具进行开发，并使用 Xilinx ISIM 工具进行了仿真验证，仿真结果表明该协议栈设计能够正确完成 ARP、ICMP、IPv4、UDP 等协议报文的发送和接收等功能。如下图 7 所示，协议栈实现了 10G 速率下数据线速的接收和发送功能。

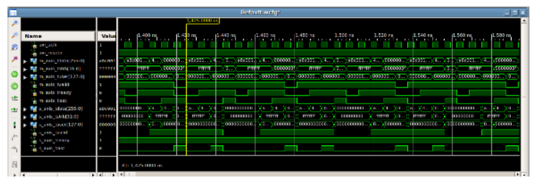


图 7　功能测试结果

Fig. 7　Functional test results

为了对本文所提出的设计方案进行性能验证，采用斯博伦 TestCenter3.61 的两个 10G 接口进行实际发包测试，采用逐步增大测试数据包 MTU 长度的方法，测试本系统以及基于 LINUX 原生 UDP 协议栈在处理不同数据包长情况下的性能，测试结果如图 8 所示。

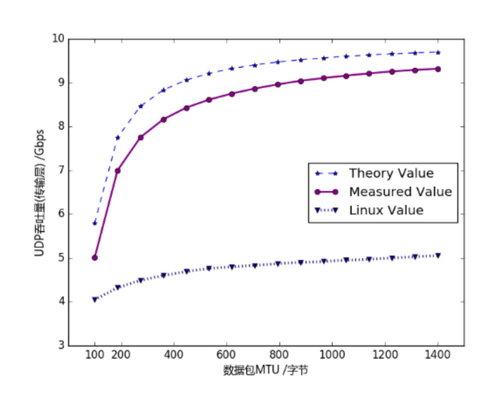



图 8 性能测试结果

Fig. 8　Performance test results

从图 8 中看以看出，随着数据报文 MTU 值的增加，硬件协议栈对于 UDP 协议负载的传输能力越发贴近理论值，而软件实现的 UDP 协议栈限于处理器串行处理能力的瓶颈，与本方案的性能差距愈发明显；当 MTU 达到 1400 字节时，协议栈对于 UDP 数据可以达到 9.32Gbps 的传输速率，接近理论值 9.7Gbps，达到了设计的要求，满足 10G 链路上线速传输数据的需求。

## 3 结束语

传统基于 FPGA 的 UDP/IP 协议栈研究都是集中在一对一的点到点传输，无法适用于 10G 网络环境下提供多点并发连接的服务器场景，本文研究了一种面向服务器应用场景的 UDP 硬件协议栈，本设计充分利用硬件在并行处理上的优势，流水化设计 UDP 协议处理流程，将 UDP 协议处理固化在 FPGA 中，降低于传输层协议对处理器的处理需求，提升系统的传输效能，通过实验结果表明，该协议栈可以适用于 10G 网络 UDP 协议的线速处理。在未来的工作中，将面向 5G 应用场景上一些传输速率非常高的服务进行研究和讨论，并研究其与智能网卡方向的结合与应用。

## 参考文献：


[1] 柯洋. 基于 FPGA 的高速数据传输板设计与开发 [D]. 武汉: 华中师范大学, 2020.

[2] 熊光阳, 王野, 李志茹, 等. 基于 FPGA 的千兆 UDP/IP 协议栈的实现及其在高速图像传输中的应用 [J]. 仪器仪表用户, 2020, 27 (03): 38-41.

[3] 崔鹤, 刘云清, 盛家进. 基于 FPGA 的 UDP/IP 协议栈的研究与实现 [J]. 长春理工大学学报 (自然科学版) , 2014 (2): 133-137.

[4] Guixé Orriols G. Design and implementation of an UDP/IP Ethernet hardware protocol stack for FPGA based Systems [D]. Barcelona: Universitat Politècnica de Catalunya, 2019.

[5] 冯琛. 基于 SAR 的高性能协议处理引擎技术研究 [D]. 杭州: 浙江大学, 2021.

[6] Nianyun Liu, Zhiqiang Xu. The Design of High-Speed Hardware UDP/IP Stack Based on FPGA for Large-Scale Sensing Systems. Journal of Internet Technology, 2017, 18 (3): 579-587.

[7] 王禹衡. 基于 FPGA 的 10G 以太网 UDP/IP 处理器视频传输接口设计 [D]. 沈阳: 沈阳工业大学 2018.

[8] 夏杨. 基于 FPGA 的万兆以太网数据分发平台设计 [D]. 北京: 北京理工大学 2016.

[9] 彭鹏. 基于万兆以太网的核物理实验高速数据传输系统研究 [D]. 兰州: 西北师范大学, 2021.

[10] https://www. plda. com/products/fpga-ip/xilinx/fpga-ip-tcpip/quicktcp-xilinx/.

[11] Langenbach U, Berthe A, Traskov B, *et al.* A 10 GbE TCP/IP hardware stack as part of a protocol acceleration platform [C]// Proc of 2013 IEEE Third International Conference on Consumer Electronics. Berlin (ICCE-Berlin) . IEEE, 2013: 381-384.

[12] http://www. intilop. com/tcpipengines. php/.

[13] http://www. dinigroup. com/new/TOE. php/.

[14] 刘宇寒. 基于 NetFPGA10G 的网络数据包加密实现及其低功耗研究 [D]. 杭州: 杭州电子科技大学, 2018.